SNU 4541.664A Program Analysis, Spring 2009
Final Exam

06/15/2009, 19:00-22:00

Problem 1 (5 pts) 아래 기초적인 분석 알고리즘의 빈 칸을 메꾸시오

$$
\begin{array}{l}
\mathit{Tabulate}(\hat{\mathcal{F}}\colon (\mathit{Code} \to \hat{D}) \to (\mathit{Code} \to \hat{D}),\ C\colon \mathit{Code}) \\
T, T'\colon \mathit{Code} \to \hat{D}; \\
\texttt{begin} \\
\quad \forall C_i \text{ of } C : T(C_i) := T'(C_i) := \bot_{\hat{D}}; \\
\quad \texttt{repeat} \\
\quad\quad T' := T; \\
\quad\quad \forall C_i \text{ of } C : T(C_i) := \boxed{} \\
\quad \texttt{until}\ \ T \sqsubseteq T' \text{ (* no more increase *)} \\
\texttt{end}
\end{array}
$$

Problem 2 (10 pts) 아래 분석 알고리즘의 빈 칸을 메꾸시오. 넓히기(widening)과 좁히기(narrowing)를 사용해야 하는 경우이다.

$$
\begin{array}{l}
\mathit{Tabulate}_{\bigtriangledown}^{\bigtriangleup}(\hat{\mathcal{F}}\colon (\mathit{Code} \to \hat{D}) \to (\mathit{Code} \to \hat{D}),\ C\colon \mathit{Code}) \\
T, T'\colon \mathit{Code} \to \hat{D}; \\
d\colon \hat{D}; \\
\texttt{begin} \\
\quad \forall C_i \text{ of } C : T(C_i) := T'(C_i) := \bot_{\hat{D}}; \\
\quad \texttt{repeat} \\
\quad\quad T' := T; \\
\quad\quad \forall C_i \text{ of } C : \\
\quad\quad\quad d := \hat{\mathcal{F}}\ (\lambda x.T(x))\ C_i; \\
\quad\quad\quad \boxed{} \\
\quad \texttt{until}\ \ T \sqsubseteq T' \text{ (* no more increase *)} \\
\quad \texttt{repeat} \\
\quad\quad T' := T; \\
\quad\quad \forall C_i \text{ of } C : T(C_i) := \boxed{} \\
\quad \texttt{until}\ \ T' \sqsubseteq T \text{ (* no more decrease *)} \\
\texttt{end}
\end{array}
$$

Problem 3 (10 pts) 아래 분석 알고리즘의 빈 칸을 메꾸시오. 할일만 하기(worklist) 방식이고, 넓히기(widening)과 좁히기(narrowing)를 사용할 필요가 없는 경우이다.

$Tabulate(\hat{\mathcal{F}}: (Code \to \hat{D}) \to (Code \to \hat{D}),\ C: Code)$
$T: Code \to \hat{D}, \quad y: \hat{D}, \quad W: 2^{Code}, \quad w: Code$

$f(c: Code): \hat{D}$
```
begin
    record that evaluation of w requires that of c;
    return T(c)

begin
    ∀C_i of C : T(C_i) := T'(C_i) := ⊥_D̂;
    W := {C_i | C_i ∈ C}
    repeat
        w := Select(W)
        y := [          ]
        if [          ] then
            T(w) := y
            ∀w' whose evaluation needs that of w :
                W := Add(W, w')
    until W = {}
end
```

Problem 4 (20 pts) 다음의 언어로 정의되는 프로그램의 요약해석을

$$e \ \to \ z \ | \ e\texttt{+}e \ | \ \texttt{-}e \ | \ \texttt{if}\ e\ e\ e$$

아래의 요약공간

$$2^{\mathbb{Z}} \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} \hat{A} = \{\bot, \texttt{+}, \texttt{-}, \texttt{0}, \texttt{0+}, \texttt{-0}, \top\}$$

에서 정의하고 그 정의가 실제 의미를 모두 포섭한다는 것을 요약해석의 틀에서 증명하라.

$$\begin{array}{rcl}
\gamma\,\bot & = & \emptyset \\
\gamma\,\texttt{0} & = & \{0\} \\
\gamma\,\texttt{-} & = & \{z \in \mathbb{Z} \mid z < 0\} \\
\gamma\,\texttt{+} & = & \{z \in \mathbb{Z} \mid z > 0\} \\
\gamma\,\texttt{-0} & = & \{z \in \mathbb{Z} \mid z \leq 0\} \\
\gamma\,\texttt{0+} & = & \{z \in \mathbb{Z} \mid z \geq 0\} \\
\gamma\,\top & = & \mathbb{Z}
\end{array}$$

Problem 5 (85 pts) Consider the following imperative language C--:

$$\begin{array}{llcl}
\text{program} & pgm & \to & c \\
\text{command} & c & \to & x \ \texttt{:=}\ e \mid x* \ \texttt{:=}\ e \mid c\ \texttt{;}\ c \\
 & & \mid & \texttt{if}\ e\ \texttt{then}\ c\ \texttt{else}\ c \\
 & & \mid & \texttt{repeat}\ c\ \texttt{until}\ e \\
\text{expression} & e & \to & z \mid \text{true} \mid \text{false} \\
 & & \mid & x \mid x* \mid e\ \texttt{+}\ e \mid e\ \texttt{-}\ e \\
 & & \mid & x\ \texttt{<}\ e \mid x*\ \texttt{<}\ e \mid \texttt{malloc} \mid \texttt{readint}
\end{array}$$

Command changes the memory. Expression computes a value. Command assigns a value to a memory location denoted (x) or dereferenced (x*) by a variable, does a sequence of commands, branches based on a boolean condition, and repeats until a condition is true. Expression value is either an integer, a location, or a boolean. Expression reads an integer (`readint`) from the outside world, is a constant integer, is the value at a location denoted (x) or dereferenced (x*) by a variable, is the result of the usual integer or boolean operations, or is a freshly allocated (`malloc`) integer-sized location.

The C-- has been used to program the inertia navigation system of the Korean liquid-fuel rocket KSR-XII. The C-- program controls the KSR-XII rocket until it reaches its orbit.

Because KSR-XII's engineers have experienced many failures of the predecessor rockets soley because of software errors, this time they want to make sure that their software is completely bug-free. KSR-XII's definition of bug-freeness is:

- every integer variable must have values within particular ranges. For example, some variable that determines the rocket's throttle valve must not exceeds some limit.
- every location variable must store at most one location throughout the program execution.

Your company offered them the software technology for the problem: static analysis. Design your analyzer(25 pts) including the three semantics (standard/collecting/abstract semantics), prove that the design is correct(30 pts), and roughly show the fixpoint steps(30) for the following example programs to demonstrate its reasonable accuracy.

- Example 1

```
x := 0;
repeat
   x := x+1;
until x < 1000
```

- Example 2

```
x := malloc; x* := 0; y := x
repeat
   x* := x* - 1; y* := y* + 3
until x < 1000
```

- Example 3

```
x := malloc; x* := 1; y := x; z := 0; i := readint;
if i < 0
then x* := x* + y*; z := 1  else (x* := x* - y*; z := 3)
x* := x* + z
```

- Example 4

```
x := malloc; x* = 0; y := malloc; y* = 1; i := readint;
repeat
  if i < 0
  then y := x else x := y;
  x* := y*+1;
until x* < 10
```

END